The 1997 Petaflops Algorithms Workshop Summary Report

May 9, 1997

Introduction

This workshop was held in Williamsburg, Virginia during the week April 13-18, 1997. It was organized to address the algorithmic research questions presented by the exciting potential of future computer systems that perform at a sustained rate of one petaflops (also written one Pflop/s, i.e. 10^{15}floating-point operations per second). This rate is between 1,000 and 10,000 times faster than the most powerful systems available today. The current consensus of scientists who have performed initial studies in this field is that petaflops systems may be feasible by the year 2010, assuming that key technologies continue to progress at current rates. Such a computing capability will permit the solution of large problems projected to be required for various federal mission applications in the 2010 time frame. In addition, if past history is any indication, the availability of these new systems will enable some completely new applications that can only be dimly envisioned at the present time.

It should be emphasized that we could also use the term "peta-ops", since these systems will be required to perform intensive integer and logical computation in addition to floating-point operations. It should also be emphasized that achieving one Pflop/s is not a goal being sought for its own sake, but as emblematic of the general need to greatly accelerate the solution (i.e. to reduce the run time) of future science and engineering problems.

Indeed, "petaflops computing" has come to mean more than merely the future goal of one Pflop/s computing rate. It has come to refer to a growing body of research dealing with the issues of very highly parallel computing. This derives from the consensus of scientists in the field (see the various federally sponsored "point design" studies) that petaflops computers will have between 10,000 and 1,000,000 processors, along with deep multi-level memory hierarchies. In fact, it is clear from the technology trends behind these predictions that future computing technology in general will be characterized by highly parallel, hierarchical designs. Thus research on petaflops systems will pave the way for the solution of similar issues that are bound to arise in mainstream scientific and even non-scientific computing.

The design and implementation of efficient algorithms for such systems present unprecedented challenges to computational scientists and engineers. It is also clear that these challenges need to be addressed now, so that the underlying principles will be well understood, and sample implementations in hand, when prototype petaflops systems are available. In addition, algorithm research performed in the next few years will provide

invaluable feedback to the hardware technologists, system architects and system software scientists involved in the design of these systems. Further, it is hoped that research in algorithms for petaflops systems will result, in at least some cases, in schemes that are fundamentally faster on a wide class of scientific computer systems, not just on future petaflops systems.

Principal Findings and Recommendations

The overall conclusion of the workshop participants is that petaflops computing is feasible, in the particular sense that at least some of the key algorithms now used on high performance computing systems do appear to be scalable to petaflops systems. Other algorithms are potentially scalable, but significant research challenges need to be overcome. Some classes of algorithms appear unsuitable for petaflops architecture. However, further research may lead to alternate algorithms to solve the same problems that are more suitable to petaflops systems. Moreover, if history is any guide, algorithm research addressing the challenges of petaflops computing is likely to yield, in at least some cases, fundamentally superior algorithms, which produce faster solutions on a wide range of computer systems. For example, a recent federal agency report found that algorithm improvements were responsible for between three and four orders of magnitude performance improvement for certain applications during the years 1970-1990 [Grand Challenges: High Performance Computing and Communications, Supplement to the President's 1992 Budget, pg. 15].

The principal findings and recommendations of the workshop participants are as follows:

Concurrency

1. Concurrency - There is concern that some of the algorithms used on today's highly parallel systems may not possess the enormous levels of concurrency (on the order of 10^8) required for the proposed petaflops designs. This conclusion holds for both the hybrid technology multi-threaded (HTMT) and the various commodity off-the-shelf (COTS) technology designs. The concurrency requirement for a particular design and algorithm might be reduced if one can assume high cache hit rates, but it will still exceed one million in most cases. The concurrency challenge is exacerbated by the pursuit of memory-thin designs, relative to the ratio of main memory to processing power that prevails in systems today. We believe that many algorithms will possess the required levels of concurrency, but since few analyses of this sort have been published, it is hard to know for sure. Scaling studies are thus needed to better understand this issue.

Data locality

2. Data locality - Although programmers of highly parallel systems have developed considerable expertise in finding and exploiting data locality in various algorithms, few studies have been published with quantitative measures of data locality. There is not even a good metric of data locality in the current literature. As a result, it is difficult to determine whether various algorithms will run efficiently in the deep memory hierarchies proposed in the petaflops designs. Thus research is needed in developing effective methodologies for assessing and exploiting data locality.

Latency and bandwidth

3. Latency and bandwidth - The latencies that can be tolerated by algorithms at various levels of granularity are not well understood, and so analyses are needed to better understand the extent to which the relatively high latencies proposed in the petaflops

designs will limit achievable performance. A related issue is synchronization. The potential impact of relatively high synchronization times on algorithm performance needs further study.

Along this line, it was noted that one potential means to hide latency is to employ algorithms, such as transpose-based methods, that exploit high-level parallelism. These schemes generally require high system bandwidth. Other algorithms can hide latency by utilizing low-level parallelism. These schemes generally require hardware features such as prefetch queues and multithreading. Thus there are intriguing opportunities here for innovative algorithms, but the hardware capabilities must be planned and sufficiently understood to push the algorithm development and evaluation.

Numerical accuracy

4. Numerical accuracy - The accuracy and stability of numerical methods as problems are scaled to petaflops sizes are not well understood, and there is concern that some algorithms may suffer significant losses of numerical precision. For some algorithms, computational scientists may need to choose between faster but less stable schemes and slower but more accurate schemes. In any event, the scaling numerical behavior of various algorithms is worthy of more careful study.

Along this line, it appears that 64-bit floating-point arithmetic will be inadequate for some applications by the 2010 time frame. Thus petaflops systems will need to provide some hardware support for 128-bit floating-point arithmetic (this conclusion was also reached in at least one previous petaflops workshop). In addition, some classified applications would benefit from 128-bit integer arithmetic. There are some applications for which even 128-bit arithmetic is not sufficient. However, these applications typically utilize multiple precision arithmetic software, such as the facility provided in Mathematica. Since such applications are still relatively rare, and since for the time being they can utilize multi-precision software designed to run on standard IEEE hardware, it appears premature at this time to consider hardware support for these extreme levels of precision.

New algorithmic approaches

5. New algorithmic approaches - There is reason to believe that new algorithms and related techniques may exist that are fundamentally more appropriate for petaflops computing than existing algorithms. In particular, there may exist algorithms that are superior in at least one of the above four categories: concurrency, data locality, latency tolerance and numerical stability. Some possibilities along this line were mentioned during this workshop. However, additional research is required to assess these alternate approaches. The intent here is not to duplicate general algorithm research already being conducted in the computational science field, but rather to focus on issues specific to future petaflops-class systems.

Unstructured grid methods

6. Unstructured grid methods - Unstructured grid methods were identified as a particularly strategic area of algorithm research and development, since these methods are likely to be utilized in a wide variety of future high-end applications. However, the fundamental scalability of these schemes, particularly for those that involve dynamic mesh refinement and dynamic load balancing, is not well understood. More research thus is needed into refining these methods and analyzing their scalability to petaflops-scale systems.

Cache Coherence

7. Cache Coherence - Currently some systems provide global cache coherence, while others do not. Some in the workshop who have hardware expertise noted that global cache coherency will be very hard to achieve in a petaflop system. Thus research is needed on algorithms that do not require global cache coherence.

Fault tolerance

8. Fault tolerance - Systems of this scale and novelty are likely to exhibit significantly higher rates of software and hardware failure. Thus special hardware and software facilities are needed to insure reliable computation (this conclusion was also reached in previous workshops). One idea mentioned in this workshop is to design algorithms and implementations that permit easy recovery from system failures. The extent to which this can be done is worthy of some study. Any analyses in this area should be coordinated with the ongoing studies of hardware mechanisms to detect and mitigate errors.

Detailed performance analyses

9. Detailed performance analyses - Much may be learned about the potential scalability of various algorithms, with regard to the issues mentioned above, using relatively simple analyses (see the algorithm case studies, to be published). However, elementary analyses of this sort cannot possibly determine with certainty that an algorithm will scale successfully to petaflops levels. To gain greater confidence, more detailed studies, including quantitative analyses of memory hierarchy operation, are required. These analyses will likely involve system simulations, both at the node level and at the network level. The most accurate simulations will themselves require highly parallel computer systems to run in reasonable time.

Along this line, the workshop participants identified a critical need for improved performance monitoring facilities, hardware and software, in future systems (this conclusion was also reached at previous petaflops workshops). For example, at present we do not even have adequate performance tools for analyzing single processor systems. Accurate timers and operation counters, for instance, are not available on many systems. Compiler feedback on performance issues is scarce, as are tools to permit the programmer to better understand cache behavior. Analyzing the performance of a program running on a system with seven levels of memory hierarchy, and with 100,000+ nodes, will be an impossibility unless very powerful capabilities are provided.

Algorithm improvement metrics

10. Algorithm improvement metrics - It was noted during the workshop that there are no rigorous metrics (except for raw operation count) to measure advances in algorithms apart from improvements in computer technology. In other words, there is no accepted methodology to assess algorithm improvements, such as hierarchical designs, that result in faster run times on a wide variety of modern systems. It is increasingly recognized that out-of-cache loads and stores, more than multiplies and square roots, are the instructions on the critical path of program execution. An effective and meaningful methodology for measuring algorithmic improvements that go beyond the flop count and reward cache locality would be quite useful in the field.

An expanded algorithm research community

11. An expanded algorithm research community - The challenge of designing algorithms for future very highly parallel computers suggests the need to involve other segments of the research community, such as the mathematical research community, in this effort. Possible areas of investigation along this line include: (1) Monte Carlo approaches to combinatorial problems; (2) conflict graphs to predict data motion and

compensate for latency (3) probability and queueing theory techniques to analyze networks; and (4) perfect hash functions to save on data communication requirements; and (5) formal methods to certify the correctness of petaflops algorithms and hardware logic designs. It may be appropriate to form special multi-disciplinary teams devoted to algorithms for future systems.

New languages and constructs

12. New languages and constructs - Given the difficulty in developing high-performance implementations of advanced algorithms on today's highly parallel systems, the possibility of developing a new language formalism for highly parallel computing, as an alternative to HPF and MPI, should be considered. One possibility here is to examine more carefully alternate languages that are already being used in some quarters of the high performance computing community. In any event, new language constructs need to be provided to existing languages (this conclusion was also reached at previous workshops). The most important of these is improved facilities to specify and control (to a limited degree) the placement and movement of data through the memory hierarchy (i.e. "spatial" and "temporal" locality). Current languages, including HPF and MPI, are considered inadequate in this regard. Improved facilities for expressing parallelism (task parallelism as well as data parallelism) at various levels should also be pursued.

Numerical library routines

13. Numerical library routines - Given the difficulty of implementing highly optimized implementations of key algorithms on these systems, it is important to facilitate the greater availability and increased usage of library routines. One possibility here is to explore standard conventions for data layout, so that it is easier to develop routines that can be widely used.

Preliminary Assessments of Algorithm Scalability

The participants in the workshop made some preliminary assessments of algorithm scalability, as given in the list below. This list is not intended to be a comprehensive list of algorithms for petaflops applications. It is limited to those algorithm areas that workshop participants felt sufficiently expert to assess.

These algorithms are divided into three categories: (1) those which appear to be scalable to petaflops systems, given appropriate effort; (2) those which appear scalable, provided certain significant research challenges are overcome; and (3) those which appear to possess major impediments to scalability, from our present perspective. These assessments are only preliminary judgments, based in most cases on experiences with current parallel implementations, as well as some cursory analysis of algorithm parameters. It is hoped that these assessments can be sharpened in the coming months and years as the issues of very highly parallel computing are further explored.

These algorithms are divided into three categories: (1) those which appear to be scalable to petaflops systems, given appropriate effort; (2) those which appear scalable, provided certain significant research challenges are overcome; and (3) those which appear to possess major impediments to scalability, from our present perspective. These assessments are only preliminary judgments, based in most cases on experiences with current parallel implementations, as well as some cursory analysis of algorithm parameters. It is hoped that these assessments can be sharpened in the coming months and years as the issues of very highly parallel computing are further explored.

For the purposes of these assessments, a petaflops system problem is defined as a problem that cannot be solved on smaller machines, or a conventional network of smaller machines, due to

- Large memory requirements
- Large computational requirements
- Large internode communication requirements
- Small internode latency requirements

In addition, the problem should run on a petaflops system in a reasonable time.

Scalable with appropriate effort:

- Dense linear algebra algorithms
- FFT algorithms (provided system has sufficient global bandwidth)
- Partial differential equation solvers for statically gridded problems, including explicit and implicit schemes
- Sparse symmetric direct solvers, including positive definite and indefinite methods
- Sparse iterative solvers (provided the preconditioner is parallelizable)
- Tree code algorithms for n-body problems
- Monte Carlo algorithms for quantum chromodynamics calculations
- Radiation transport algorithms
- Certain highly concurrent classified algorithms

Scalable provided significant research challenges are overcome:

- Dynamic unstructured grid methods, including mesh generation, mesh adaptation and load balancing
- Molecular dynamics algorithms
- Interior point-based linear programming methods
- Data mining algorithms, including associativity rules, clustering, and similarity search schemes
- Sampling-based optimization and search techniques, including genetic algorithms
- Branch and bound search algorithms
- Boundary element algorithms
- Symbolic algorithms, including Grobner basis methods
- Discrete event simulation
- Certain classified algorithms that involve random memory accesses

Possessing major impediments to scalability:

- Sparse unsymmetric Gaussian elimination
- Theorem proving algorithms
- Sparse simplex linear programming algorithms

- Integer relation and integer programming algorithms

A Success Metric

More than one participant emphasized the fact that many, if not most, petaflops system jobs are likely to consist of an aggregate of almost embarrassingly parallel, modest-sized calculations, such as to explore the parameter space for aircraft design optimization. Even today, such usage comprises a large fraction of total system cycles on many large scientific systems. Indeed, such jobs are usually the production-oriented calculations that are the basis for a center's funding. "Heroic" calculations that press the limits of available technology are often done by researchers who lack such strong financial support. However, most large scientific centers also recognize that these heroic jobs are needed to lead the way into future high performance computing applications. In short, there is a well-recognized symbiotic relationship between modest-sized production usage and heroic, research usage.

In spite of these considerations, the participants felt that if petaflops systems are only effective on aggregates of modest-sized calculations, then their success (as well as the success of a research program in petaflops algorithms) would be questionable. One reason for this judgment is the observation that such applications may run on networks of RISC symmetric multiprocessors of comparable technology generation.

In other words, the consensus of the workshop participants is that a rigorous success metric for petaflops computing R&D must involve at least some true petaflops system problems (in the sense described above). Along this line, we suggest the following as a possible metric of the success of a program in petaflops algorithms: if all of the algorithms listed in the first category above, and at least one in the second category, have been successfully implemented in applications running on petaflops system, with sustained performance rates of at least 0.1 Pflop/s, then the research effort in petaflops algorithms has been successful.

It is hoped that the above findings and recommendations will form the structure of a rigorous research agenda in the field.